DEPARTMENTOF
COMPUTERSCIENCE&ENGINEERING

Discover. Learn. Empower.

Experiment 2

Name: Krishan dev ojha UID: 22BCS15456

Branch: BE-CSE Section/Group: 22BCS_IOT-635-B

Semester: 6th Date of Performance:20/01/25

Subject Name: Project Based Learning in Java with Lab

Subject Code: 22CSP-359

1. Aim: The aim of this project is to design and implement a simple inventory control system for a small video rental store. Define least two classes: a class Video to model a video and a class VideoStore to model the actual store. Assume that an object of class Video has the following attributes:
   1. A title;
   2. a flag to say whether it is checked out or not;
   3. An average user rating.

   Add instance variables for each of these attributes to the Video class.

   In addition, you will need to add methods corresponding to the following:

   1. being checked out;
   2. being returned;
   3. receiving a rating.

   The VideoStore class will contain at least an instance variable that references an array of videos .

   The VideoStore will contain the following methods:

   1. addVideo(String): add a new video (by title) to the inventory;
   2. checkOut(String): check out a video (by title);
   3. returnVideo(String): return a video to the store;
   4. receiveRating(String, int) : take a user's rating for a video; and 5. listInventory(): list the

   whole inventory of videos in the store.

2. Objective: Create a VideoStoreLauncher class with a main() method which will test the functionality

   of your other two classes. It should allow the following.

   1. Add 3 videos: "The Matrix", "Godfather II", "Star Wars Episode IV: A New Hope".
   2. Give several ratings to each video.
   3. Rent each video out once and return it.

List the inventory after "Godfather II" has been rented out.

3. Algorithm:

- Initialization: Create the Video class to represent a video with attributes: title, checked-out status, average rating, and rating count.

  Create the VideoStore class to manage an inventory of videos with a fixed capacity.

  Define methods in Video for checkout, return, adding ratings, and accessing details.

  Define methods in VideoStore for adding videos, finding videos, managing checkouts, adding ratings.

- Main Execution:

  Create an instance of VideoStore with a maximum capacity of 10 videos.

  Add three videos to the store using the addVideo() method.

  Provide ratings to the videos by calling receiveRating() for each title with a rating.

- Perform the following operations:

  Check out the video "Godfather II" using checkOut().

  Return the video "Godfather II" using returnVideo().

  List the current inventory of the store by calling listInventory().

- Supporting Steps:

  addVideo(): Check if inventory space is available; if yes, create a new Video object and store it.

  receiveRating(): Locate the video by title using findVideo(), and update its average rating.

  checkOut(): Mark the video as checked out if it is not already checked out. returnVideo(): Mark the video as returned if it is currently checked out. listInventory(): Iterate over all videos in the store and display their details.

4. Code:

```
class Video { private String
title; private boolean
checkedOut; private double
```

```
averageRating; private int
ratingCount; public
Video(String title) {
this.title = title; this.checkedOut
= false; this.averageRating =
0.0; this.ratingCount = 0;
} public void checkOut()
{   if   (!checkedOut)   {
checkedOut = true;
System.out.println("Video \"" + title + "\" has been checked out.");
} else {
System.out.println("Video \"" + title + "\" is already checked out.");
} } public void
returnVideo() { if
(checkedOut) { checkedOut
= false;
System.out.println("Video \"" + title + "\" has been returned.");
} else {
System.out.println("Video \"" + title + "\" was not checked out.");
} } public void receiveRating(int rating)
{ if (rating < 1 || rating > 5) {
System.out.println("Invalid rating. Please rate between 1 and 5."); return;
} averageRating = (averageRating * ratingCount + rating)
/
(++ratingCount);
System.out.println("Received rating of " + rating + " for video \"" + title +
"\"."); } public String
getTitle() {
return title; } public boolean
isCheckedOut() {
return checkedOut;
}
public double getAverageRating() { return
averageRating;
}
} class

VideoStore { private

Video[] videos;

private int count;

public

VideoStore(int
```

```
capacity) { videos =

new

Video[capacity];

count = 0; } public void addVideo(String
title) { if (count < videos.length) { videos[count++]
= new Video(title);
System.out.println("Added video: " + title);
} else {
System.out.println("Inventory is full. Cannot add more videos.");
} } public void checkOut(String title)
{ Video video = findVideo(title); if
(video != null) { video.checkOut();
} else {
System.out.println("Video \"" + title + "\" not found.");
} } public void returnVideo(String title)
{ Video video = findVideo(title); if
(video != null) { video.returnVideo();
} else {
System.out.println("Video \"" + title + "\" not found.");
} } public void receiveRating(String title, int
rating) { Video video = findVideo(title); if (video
!= null) { video.receiveRating(rating);
} else {
System.out.println("Video \"" + title + "\" not found.");
} } public void listInventory() {
System.out.println("\nInventory:"); for
(int i = 0; i < count; i++) {
Video video = videos[i];
 System.out.println("Title: " + video.getTitle() + ", Checked Out: " + video.isCheckedOut()
+
", Average Rating: " + video.getAverageRating());
} } private Video findVideo(String
title) { for (int i = 0; i < count; i++) {
if (videos[i].getTitle().equalsIgnoreCase(title)) {
return videos[i];
} } return
null;
}
}

public class VideoStoreLauncher {
```

```
public static void main(String[] args) { VideoStore
store = new VideoStore(10); store.addVideo("The
Matrix"); store.addVideo("Godfather II");
store.addVideo("Star Wars Episode IV: A New Hope");
store.receiveRating("The Matrix", 5);
store.receiveRating("Godfather II", 4);
store.receiveRating("Star Wars Episode IV: A New Hope", 5); store.checkOut("Godfather
II");
store.returnVideo("Godfather II");
store.listInventory();
}
}
```

5. Ouput:

```
Added video: Godfather II
Added video: Star Wars Episode IV: A New Hope
Received rating of 5 for video "The Matrix".
Received rating of 4 for video "Godfather II".
Received rating of 5 for video "Star Wars Episode IV: A New Hope".
Video "Godfather II" has been checked out.
Video "Godfather II" has been returned.

Inventory:
Title: The Matrix, Checked Out: false, Average Rating: 5.0
Title: Godfather II, Checked Out: false, Average Rating: 4.0
Title: Star Wars Episode IV: A New Hope, Checked Out: false, Average Rating: 5.0
PS C:\Users\Vivek\Desktop\java>
```

6. Learning Outcomes:

o Understand how multiple classes work together in a structured application. o Learn to add, manage, and interact with objects dynamically using arrays. o Practice input validation (e.g., ratings) and handling state-based conditions like checking out videos.